Multi-Agent Active Collaboration Between Drivers and Assistance Systems

Jean-Paul Barthès, Philippe Bonnifait

Multi-Agent Active Collaboration between Drivers and Assistance Systems

Jean-Paul Barthès[1,2], Philippe Bonnifait[1,2]
[1]*Université de Technologie de Compiègne (UTC)*, [2]*CNRS Heudiasyc UMR 6599, France*

***Abstract*—Intelligent vehicles refer currently to vehicles able to drive autonomously or able to provide pertinent information to the driver for safety, assistance and comfort. Cognitive cars are intelligent vehicles with additional capabilities like being able to collaborate with the driver in operating conditions. In this paper, a multi-agent system is used as a "digital butler" that does the interface between the driver and the machine. In order to test this approach, we consider an Advanced Driving Assistance System (ADAS) providing speed warnings when approaching dangerous areas. The system has been tested on an actual use case carried out with an experimental vehicle. We report some illustrative collaboration between the driver and the machine.**

***Index Terms*—Advanced Driving Assistance Systems, Multi-Agent systems, Active collaboration between driver and ADAS**

I. Introduction

Advanced Driver Assistance Systems (ADAS) are systems intended to help the driver in his driving activities. Technological solutions are many, like Adaptive Cruise Control (ACC), Intelligent Speed Adaptation (ISA) or Collision Warning Systems (CWS). When designed with a safe Human-Machine Interface (HMI), an ADAS should increase car safety and comfort.

Building a safe HMI requires careful attention as argued by ergonomics. For instance, in [1] the authors have proposed guidelines to display information to the driver in running conditions. In this kind of problem, much effort is devoted to the choice of an efficient and safe strategy to display the information coming from the perception system of the vehicle or from cooperative perception thanks to communication devices. Here, the driver is uniquely receiving information from the warning system. It is up to him to take into account the warning messages.

Some ADAS systems act in a completely different way, directly onto the control inputs of the vehicle. Obstacle detection systems like the one presented in [2] can take the decision to brake, if the situation is estimated "very" dangerous. This kind of system is usually classified as "active safety". By some ergonomists, ADAS of this kind are called "dead driver systems" [3].

For several years, people focused on systems operating between these two extremes. Such approaches are sometimes referenced as "cognitive." For instance see [4][5]. Here, the problem is to devise a closer collaboration between the driver and the machine. Monitoring the driver's activities is a first key prerequisite [6] but the problem goes far beyond. A collaboration can take place between the human and the machine to modify the setting of the parameters of the ADAS for instance. A basic example is the problem of giving the destination address to the navigation system in order to compute a route. The collaboration can also occur during the operation, while driving. This is typically the issue we consider is this paper, by proposing to use a multi-agent system (called OMAS in the following) that acts as an interface between the driver and the ADAS, a real-time platform, PACPUS, managing sensors, collecting data from the vehicle and implementing the ADAS function. By using speech recognition, the system presented here is able to understand sentences relative to the tuning of warning messages due to speeding in dangerous situations.

The remaining of paper is organized in five sections. We start by studying the classical role of multi-agent architectures for intelligent control in robotics. We present afterward a use case in which an ADAS system collaborates actively with the driver at a cognitive level. Then, we detail the system that has been developed for this purpose. We report a critical review of the results that we obtained and we conclude by presenting future extensions of this research.

II. Multi-agent architecture for intelligent control

Multi-agent architectures provide an efficient framework to implement high-level, flexible and modular control strategies, particularly for distributed and collaborative systems like for instance the control of urban traffic in large cities [7]. Some works have shown that they are also useful for autonomous systems. An example is the system used by [8] for making autonomous a vehicle at the DARPA Urban Challenge. Here, a Multi-Agent System (MAS) is used for the environment perception and the choice of adequate vehicle's behavior. Several observers and several controllers run in parallel and are triggered by the MAS. Such a mechanism is called "reactive MAS" (Figure 1). Usually, such a high level control gets along with a real-time low-level architecture in charge the feedback loops involving the actuators. Indeed, one has to make a distinction between the possibility of reasoning and low-level control. The reasoning system is rather slow with respect to the answer time needed for taking over the control of the car. When interacting with the driver, the cognitive task can be done in collaboration between the car and the driver. In this case, a third key component has to be considered explicitly. This is the "pilot assist" displayed on Figure 1. At this level, the reaction time is much longer since an active collaboration between the machine and the driver is necessary. Such an

approach is somewhat similar to the situation in warships where the real-time defense system is automatic and fast, and the advising command system is much slower.

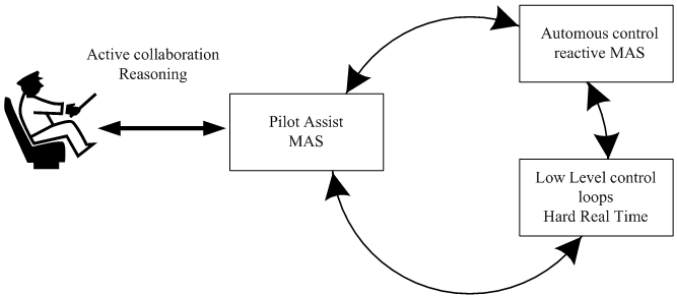

Figure 1. Possible use of Multi-Agent Systems (MAS) for automotive intelligent control

In the rest of this paper, we focus on the active collaboration between the driver and an intelligent vehicle equipped with ADAS functions.

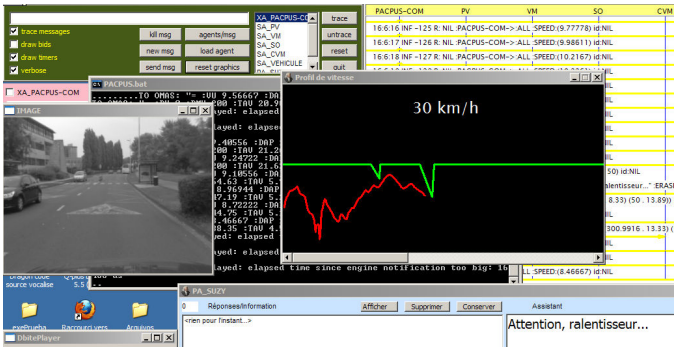

Figure 2. Alert due to bump just following a pedestrian crossing (the top left window is the OMAS multi-agent control panel, the top right window contains a graphics trace of the exchanged messages, the left window showing the road comes from the PACPUS system, the right window showing the graph is posted by Suzy, the bottom (tip of the iceberg) window is the Suzy interface window, the center background window is the Java trace of PACPUS)

III. Use Case

Let us first introduce a use case: we are in town (Compiègne) with a default speed limit of 50 km/h and want to drive up to "Lycée Charles de Gaulle," a particular high school. On the way, we encounter pedestrian crossings, bumps to slow the vehicle; we drive in front of the swimming pool before arriving to the high school. Fig. 2-5 are screen captures from the experiment. On the figures you can see several windows: On the left, an obvious image showing the street, next to it a graph plotting the actual vehicle speed vs. the maximal allowed speed, on the bottom right part of the interaction screen between the driver and his personal assistant named SUZY.

Fig. 2 shows that the car is driving well under the speed limit. A first notch in the top profile indicates that the speed should be reduced (because of a pedestrian crossing area), immediately followed by a second notch indicating that the speed should be reduced because of a bump on the pavement. The information is provided by PACPUS. The bottom right part of the screen displays the message "Attention, ralentisseur...[1]" uttered by SUZY. The announcement of the bump painted onto the pavement can be seen in the street display on the left.After that the street is free of other cars and the driver takes the car over the speed limit. A message is uttered by SUZY: "Trop vite![2]" and a red window flashes on the bottom left of the screen. Suzy also emits a short beep.

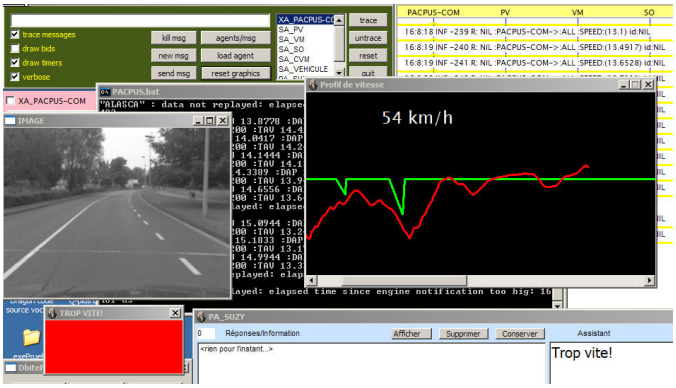

Figure 3. Speeding alert (left red window)

The next figure shows that we have reached the swimming pool, and a maximum speed profile is posted showing that the speed should be reduced significantly when driving in front of the swimming pool in this particular time.

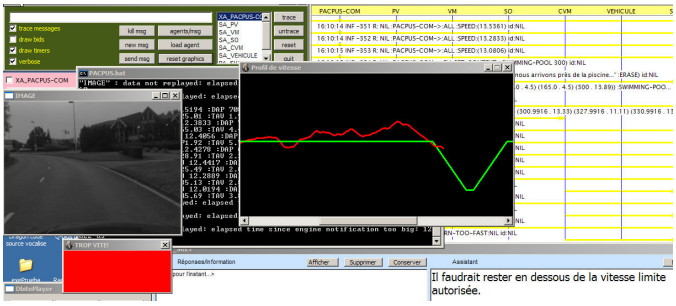

Figure 4. Swimming pool speed profile and alert

However, here the driver says "La piscine est fermée.[3]", which causes SUZY to remove the maximum speed profile and reset the maximum speed limit to 50 km/h (Fig. 5).

[1]Watch the bump...
[2]too fast!
[3]The swimming pool is closed.

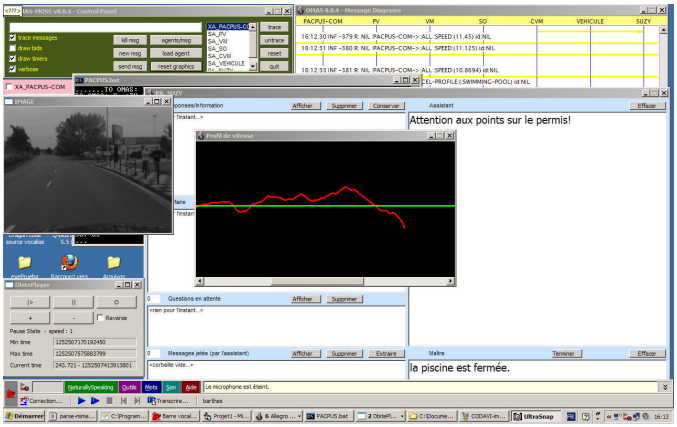

Figure 5. Swimming pool canceled profile

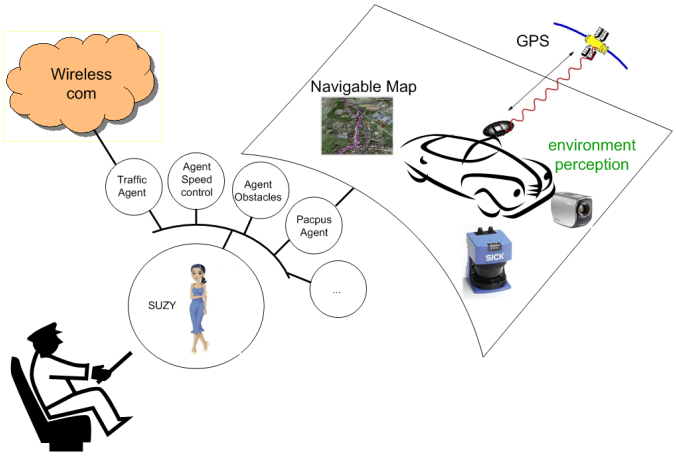

Figure 6. Multi-agent interface and organization

Then the trip continues to the high school with similar situations.

An important point is to notice that the driver can modify the behavior of SUZY by bringing new information to the system, e.g. in the swimming pool area. While speeding the driver could tell SUZY something like "Je sais.[4]" which would cause SUZY to stop sending warnings to the driver until the next obstacle is detected. The driver can thus play an active role with respect to the ADAS.

In addition, SUZY is a regular personal assistant. SUZY can be connected to external sources and give information about the traffic or the weather or any information it can get from the web, if additional agents are added to the system.

IV. Architecture

The idea behind the current research is to allow a better communication between driver and vehicle. In order to have a high-level interaction, we need to have a vehicle equipped with an ADAS able to exploit information given by the driver.

The ADAS function has been prototyped using our PACPUS system[5]. It exploits mainly a pre-calculated itinerary, a navigable map, a positioning functionality and an obstacle detection system that fuses measurements coming from a Lidar (Light Detection and Ranging) and a stereo camera system to estimate a time to collision [9].

For developing the multi-agent system, we used previous work in which we developed "intelligent" agents for driving avatars to simulate interventions in dangerous plants [10] (by coupling a multi-agent platform with a virtual reality system), to propose a model where we associate a multi-agent system with a real-time platform on-board the vehicle.

Figure 6 presents the paradigm that we consider. Every high-level information that has to be posted to the driver or that is coming from the driver has to pass through an agent called "SUZY". High level information refers here to traffic information or ADAS alerts for instance. The agent called "PACPUS" is the one in charge of the ADAS.

[4]I know.

[5]http://www.hds.utc.fr

A. The ADAS function

The ADAS function considered in this work is an ISA function: the driver is informed of speeding if the vehicle is approaching a potentially dangerous area.

This information can be addressed to the driver through an active gas pedal which for instance is made harder, a beep or a display on a dedicated screen. Figure 7 illustrates the interface between the driver and the assistant that we consider in this work.

Figure 7. Driver interface: active gas pedal, audio and screen display. The two cameras on the dashboard are used for eye-tracking.

Two kinds of information are provided by the system. The first one is said "static" and refers to what is called "Map-Enabled ADAS" meaning that the Digital Map is one of the principal inputs to the system and without it the function would not be possible. The information corresponds to points of interests (POI) that have been charted in the map, like stops, pedestrian crossings or school entrances. To implement this method, we use an "Electronic Horizon" (EH) [11] that exploits a GPS positioning map-matched to a predicted path. The ADASIS v2 protocol [12] is a standardized Application Programming Interface (API) that can be used to implement a Map-Enabled ADAS. It is important to note that EH is much

more reliable if the route has been planed before driving. The relevance of a POI alert can be modulated thanks to a calendar information. For instance, during vacation, schools are closed. This filtering issue is handled by the OMAS system.

The second kind of alert refers to "dynamic" events, like pedestrian crossing the road or traffic jams. This information is captured by the obstacle detection function based on lidar combined with camera processing. Please note that we are not exploiting this information in time critical situations, like many obstacle avoidance systems that already exist in some cars and apply a braking action in last resort. Here, this exteroceptive information is exploited as soon as possible in order to modulate the speed set-point of the ISA system.

In our experiments, we have used a system called "D-BITE" [13] to log the sensory information coming from the vehicle on the reported use cases. This system is able to collect a huge amount of time-stamped information (like different video streams). A player is then used to replay the data in a similar way to real-time conditions. This is an interesting functionality that is very useful for prototyping systems.

Figure 8 presents the vehicle that has been used in the experiments. The lidar is located in the bumper and the stereo system is visible on the rooftop. The GPS antenna is installed on the rear.

Figure 8. The experimental car used to log the sensor data

B. OMAS

OMAS is a multi-agent platform intended to develop cognitive agents. OMAS agents are fairly complex, each agent may have a number of different threads running in parallel. OMAS is described in details in [14], and the platform can be downloaded from www.utc.fr/~barthes/OMAS. In this experiment the agent platform is positioned between the user and the PACPUS platform as shown in Fig.6. The system here contains five agents: PV, VM, CVM, SO, PACPUS-COM and SUZY.

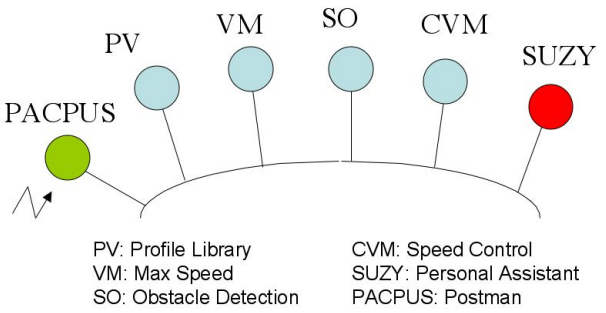

Figure 9. Architecture of the MAS

PV (Profils de Vitesse / Speed Profiles) contains a library of normalized speed profiles corresponding to various situations, e.g. "pedestrian-crossing," "bump," "swimming pool," "school," "obstacle," etc. Receiving a ADAS message triggers the choice of the corresponding profile. The profile is made dimensional and sent to the VM agent.

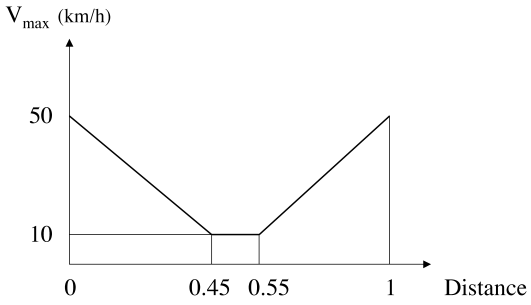

Figure 10. Maximal speed school profile

VM (Vitesse Maximale / Maximal Speed) is an agent in charge of combining profiles to compute the maximal speed that the vehicle is allowed to reach on the itinerary. Every time a new profile is computed, it is sent to the CVM agent.

CVM (Contrôle de Vitesse Maximale / Maximal Speed Control) checks that the vehicle speed stays under the speed limit (with a small tolerance when exceeding it). When the vehicle is driving over the speed limit, a message is sent to SUZY, and SUZY normally warns the driver by telling something. We have seen that SUZY may be temporarily silenced.

All messages come from PACPUS-COM, a transfer agent (also called postman), interfacing the PACPUS real-time platform with the OMAS system. PACPUS-COM is in fact a gateway between the two systems, restructures the messages and sends them to the right agent with the proper format. PACPUS-COM may use point-to-point messages or broadcast messages (e.g. for posting vehicle speed). In both cases, since we are using a UDP (User Datagram Protocol) protocol, we need a single message.

The SO (Surveillance d'Obstacle / Obstacle Detection) is linked to the vehicle obstacle detection system (lidar and stereo-vision) and indicates if there are obstacles on the road

or pedestrian walking a the sidewalk for instance. If so, it sends a message to PV that will select a new profile to be merged to the current maximal speed profile.

SUZY is a personal assistant agent in charge of presenting information to the driver through the ADAS interface and decoding vocal requests from the driver. SUZY can also answer questions about stored data and could be connected to the outside world (Web).

How does the system work?

Fig.11 shows how the multi-agent system works in the context of the experiment. At first PACPUS broadcasts a message setting up the context: we are in town (thus speed is limited to 50 km/h). The PV agent computes an indefinite flat profile limiting the speed to 50 km/h and sends it to CVM that starts controlling that the received speed does not exceed the limit (following broadcasts). At some time PACPUS indicates that we are approaching the swimming pool (information obtained from GPS data). PM sends a library profile to VM that combines it with the current profile and ships it to CVM. CVM thus finds that the vehicle is driving too fast. It then sends a warning message to SUZY. SUZY posts the message, tells it to the driver and starts an alarm. However the driver tells SUZY that he does not care or that the swimming pool is closed. SUZY sends a message to VM to remove the swimming pool profile from the combined profile. This is done by VM that sends the new combined profile to CVM. Etc.

This experiment was intended to demonstrate that the OMAS platform could be coupled to the PACPUS real-time platform. Consequently, we do not see complex reasoning. However, each agent has a personal ontology containing domain concepts like a town, a road, a swimming pool, a school, or vacations. Each agent can contain rules or methods (including demons) to assess the situation and to decide of an action. SUZY in particular should decide about a presentation policy, so that the driver is not overloaded with useless information. On the other hand, when an potential obstacle is detected, the system should decide whether alerting the driver, or, if there is not enough time to prepare the vehicle to a possible collision by pre-tensioning seat belts for example. This is possible because the PACPUS-COM agent is a gateway in both directions, meaning that messages can be sent to the PACPUS platform and action can be taken through this platform.

V. More technical Data

This section gives some additional information for technically oriented people, although the system we developed is a proof of concept and is not meant to be an actual product.

a) Hardware: The current implementation of the system doesn't exploit any exteroceptive perception system (like cameras and lidars). The ISA function uses a GPS receiver, a digital map and speed measurement coming from the CAN (Control Area Network) bus of the vehicle. The POIs considered in this work have been added manually on the navigable map.

PACPUS and OMAS software runs on different machines installed on board. For convenience the D-BITE emulator replaying logged data, the ISA function and the OMAS platform were installed on the same notebook in order to simplify software development. The connection between the PACPUS platform and the OMAS platform uses an Ethernet cable. The vocal input is done through a microphone connected to the Multi-Agent System (MAS) notebook.

Software: The PACPUS platform is based on the D-BITE software that relies on an event-triggered real-time architecture. D-BITE is written in C++ and exploits a middleware called SCOOTR [15]. SCOOTR is a fully distributed middleware based on a "client-server" approach. A hard real-time implementation has been developed with Real-Time Linux and Firewire interfaces between the computers. The system used in these experiments is soft real-time and has been implemented with MS Windows XP and Ethernet LAN.

The OMAS platform is written in Lisp (Allegro Common Lisp from Franz Lisp®). The vocal interface uses an old version of Dragon Naturally Speaking® from ScanSoft (today Nuance) and the interface was programmed using Visual Basic.

Transfers between the subsystems is done using UDP and implemented using Qt (QUdpSocket) (see Fig. 12). A specific protocol between PACPUS and OMAS was designed with a very simple content language, e.g.

"= :VU 10 :DAP 83 :TAP 5.35"

The typical fields to be used in the message are given in Table I.

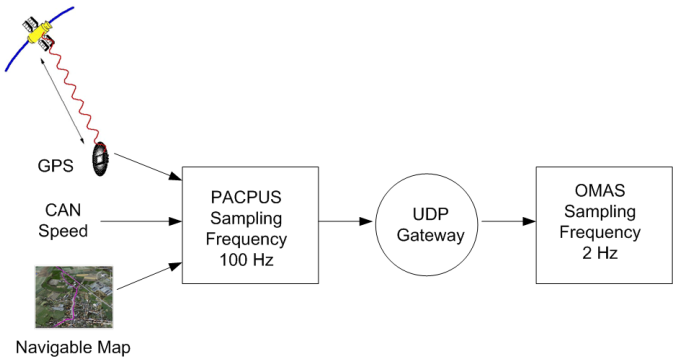

Figure 12. UDP connection between PACPUS and OMAS

Table I
FIELDS FOR THE TRANSFER CONTENT LANGUAGE

code	meaning
DAP	distance to a predefined target (e.g. school entrance)
VU	vehicle speed
TAP	time before arriving at predefined target
DV	density of obstacles
DMV	minimal distance to obstacles
TAV	time to arrive at the obstacles
ARZ	zone type, e.g. school, swimming pool... (a symbol)

It is up to the PACPUS platform to package the data and

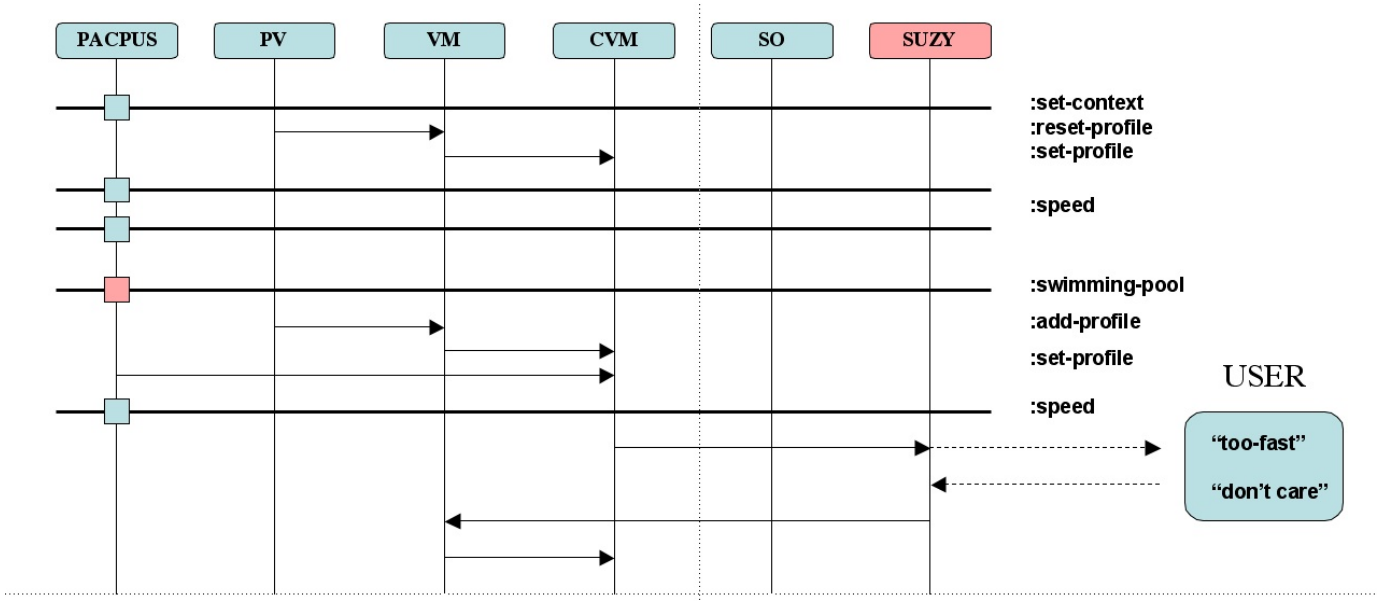

Figure 11. Agent lifelines (thicker lines represent broadcasts)

to sequence messages at a rate that can be supported by the MAS (currently 2Hz).

On the MAS side we have three types of agents: the postman receiving messages from PACPUS and also capable of sending them to the vehicle (not done in this experiment), SUZY, a personal assistant, and service agents. All agents are on the same LAN (Local Area Network) loop (here they are in the same machine, but they could be distributed). Exchanges among agents are done by UDP broadcasts. The most interesting part is the communication between the driver and SUZY.

b) Communication with SUZY: The mechanism is based on a library of tasks that SUZY knows how to do and an ontology. When SUZY receives a new input (character string), she uses a library of tasks to determine which task is the most likely to be wanted. Based on a model of each task containing linguistic cues, the system ranks the tasks in decreasing values of a task score and removes tasks below a certain threshold. The first task is then executed, resulting in sending a message somewhere (specific agent or broadcast). SUZY has an ontology describing concepts and tasks. Table II describes the concept of motorway that has two attributes: a speed limit (with a default of 130 km/h or 36.11 m/s) and a number identifying the motorway (e.g. A6).

Table II
SUZY’S CONCEPT OF MOTORWAY

defconcept	(:en “motorway” :fr “autoroute”)
:att	(:en “speed limit” :fr “vitesse limite”)(:default 36.11)
:att	(:en “number” :fr “numéro”) (:entry)

Table III describes the task for killing constraints related to a swimming pool. Linguistic cues are defined as index patterns and for each cue a weight is given allowing to compute a score for the corresponding task. A dialog reference is given, which allows triggering a subdialog associated with that task.

Table III
SUZY’S TASK FOR REMOVING SWIMMING POOL CONSTRAINTS

defindividual	“task”
:doc	:fr “Tâche d’enlèvement de la zone de la piscine”
“task name”	“remove swimming pool”
“performative”	:command
“dialog”	_remove-swimming-pool-conversation
“index pattern”	
	(:new “task-index” (“index” “piscine”)(“weight” .2)
	(:new “task-index” (“index” “fermée”)(“weight” .7)
	(:new “task-index” (“index” “pas ouverte”)(“weight” .7)
	(:new “task-index” (“index” “en grève”)(“weight” .7)

Subdialogs are modeled as finite state machines (conversation graphs). In this case the subdialog is very simple and has a single state shown [Table IV], where an answer is given to the driver and a message is sent to the VM agent. Dialogs may have any number of states and the system uses both linguistic patterns and the ontology to extract information from the user.

Table IV
SUZY’S SUBDIALOG

defstate	_rsp-entry-state
:label	“remove swimming pool dialog entry”
:entry-state	_remove-swimming-pool-conversation
:explanation	“user wants SUZY to stop sending warnings”
:text	“OK, j’ai bien noté.”
:execute	send-message
	make-instance ’omas-message :type :inform
	:from :SUZY :to VM
	:action :cancel-profile :args ’(:swimming-pool)
:transition	:success

The number of tasks that one can have in the system is not limited and it is possible and not very difficult to add as many

tasks as needed by the application. Some of the tasks can call web services if the vehicle has an Internet connection.

VI. Discussion

This section discusses the limits of the experiment presented in the paper. First, the system was implemented using a replay system of actual drives, able to replay all the data of the sensors like in real-time in the car. This has a significant advantage, namely the possibility to replay any part of the scenario as many times as necessary to test the system. The replay system can be seen on bottom left of Figure 5 and a command console helps in verifying that the computer is not overloaded, which can occur when the replay speed is high.

A. Limitations of the Current Study

Since we have been working using emulation rather than with testing the system on-board, several features have been introduced without any error. For instance, locating pedestrian crossings and "bumps" on the road or reading speed limits using on-board cameras induces inevitably false alarms and miss-detections.

1) Locating Pedestrian Crossings and Bumps: The location of pedestrian crossings and bumps was manually coded in the EH since such information is not available in the current maps of Compiègne. It could be also obtained either by receiving signals from the environment (active road-side units), or by recognition of the corresponding road signs (see Fig. 13). Currently, we have no road-side unit and no real-time program for analyzing the road signs. We think that a fusion of all these perception modalities should provide reliable information to our system.

Figure 13. Road sign for speed limit and bump

2) Recognition of the Various Areas and of the Speed Limits: The recognition of the various areas like the town of Compiègne, the swimming pool location comes again from perfect data based on global positioning and EH. In practice, recognition of speed limits should be more elaborate to take into account for instance road works. Moreover, extracting rough visual information from the picture shown in Fig. 13 is not difficult with regard to the speed sign and bump sign. It is more difficult for the intermediate annotation ("A 30m") that adds information to the above sign. The information at times is a restriction on the application of the sign, e.g. when the speed limit applies to trucks only. This should be provided either by having active road signs or by recognition by means of scene analysis.

3) Voice Recognition: There are also some issues concerning the voice recognition system. We have been using a rather old version of Dragon Naturally Speaking (version 7) in a quiet environment. The product has excellent performances in a quiet environment. It remains to be seen if this is still the case in a noisier environment. The new version of the software (version 11) sold currently by Nuance® could not be tested due to the unavailability of the corresponding SDK (programming interface). Voice interaction inside a car is a difficult problem regarding the vocal input [16]. Regarding output, the PACPUS platform however can easily redirect the output to the car radio.

B. Advantages of the Proposed Approach

The main advantages of the proposed approach in addition to the vocal interface and 2-way communications, comes from its modular architecture. Indeed, since the messages are sent in a broadcast mode, it is possible to add new features easily. Each agent will pick up the messages it needs for its particular reasoning. Some of the possible extensions are mentioned thereafter.

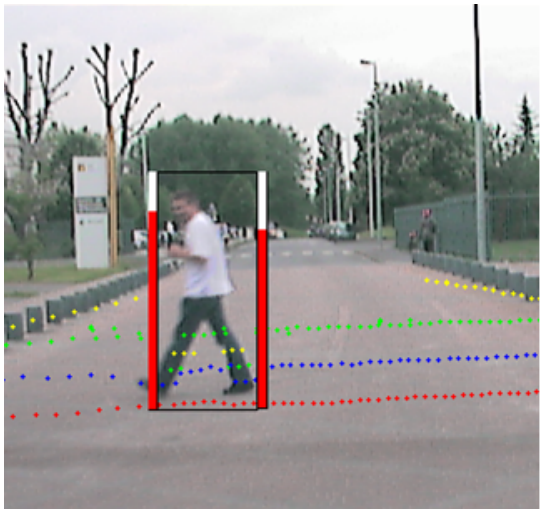

Figure 14. Pedestrian detected by a four-layer lidar. Pedestrian detection confidence level (left bar) and Pedestrian recognition Confidence (right bar)

1) Choice of the Itinerary: This feature can be implemented easily through the API interface of the navigation system. The possible dialogs are then added to the library of dialogs of SUZY, corresponding to the various information/actions that can be obtained from the navigation system. The choice of the itinerary could be the result from a discussion, rather than a simple selection of choices provided by the GPS software.

2) State of the Car: In the same manner, dialogs concerning the condition of the car itself can be added to the library. It could be used by the driver to inquire about the state of the

car or by the car to signal a specific problem. The information can be provided in real time.

3) New Sensors: e.g. Eyes Tracking: The possibility of tracking the eyes of the driver, coupled with information about the speed of the car and the environment could be used to warn the driver in case of insufficient attention to the traffic.

4) New Sensors: e.g. Detecting Pedestrians: The advanced software developed by [17] for detecting pedestrians (with confidence boxes like shown in Figure 14) coupled with dynamic information could be used to give more sophisticated warnings than when using only distance and density information.

5) Driver Profile: Driver profiles could be recorded and used to determine if the way of driving is within normal limits as given by the profile.

VII. Conclusion and Future Developments

Trying to build a cognitive car can be attempted from two different perspectives: either one tries to introduce intelligence (meaning possibilities of reasoning) in the system controlling the car, or one can try to extend knowledge based systems to introduce the necessary controls acting on the car. Both approaches are difficult and probably no so easy to develop. The approach we have chosen is to couple a system perfectly adapted to controlling the car, an ADAS implemented with PACPUS, to a system of hybrid agents that can be reactive, deliberative and collaborative. The coupling is a loose coupling done at a fairly high level. Reasoning on symbolic data requires time and the real time system must be slowed down, filtering data and transmitting them at a reasonable frequency (here 2Hz).

We have developed a prototype that works in real-time with a data player that reads sensor measurements acquired by an experimental vehicle. This prototype is a proof of concept that opens up many perspectives.

We need now to develop our approach in several directions: (i) introducing more complex reasoning taking into account a model of driver, a dynamic model of the environment, and some context; (ii) integrating the system to the vehicle hardware (e.g. using the radio or in board hardware to improve communications); (iii) integrating the prototype into a simulation system, which will allow us to run tests with different types of drivers in the laboratory rather than on the road; (iv) adding more service agents to increase features, leading to richer dialogs. The first type of improvements would introduce driver profiles and driving patterns. Integrating the system to the vehicle hardware requires some work on the side of the PACPUS platform. Currently the system is an addition to the vehicle and input and output are directly handled by the OMAS notebook. It would be nicer to use the vehicle hardware. The third improvement is important, since finding drivers to run outside test drives is not so easy, and we have a full size car simulator with additional hardware on the driver's side. Finally the last type of improvement is not very difficult to do since the multi-agent system is open and one can add agents and services at any time.

Acknowledgments

This research has been conducted under the auspices of a project called "Codavi" that regroups colleagues working in the same lab but in two different teams.

References

[1] M. Bruyas, B. L. Breton, and A. Pauzié, "Ergonomic guidelines for the design of pictorial information," *International Journal of Industrial*, 1998. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0169814196000819

[2] A. Broggi, P. Cerri, S. Ghidoni, P. Grisleri, and H. G. Jung, "A new approach to urban pedestrian detection for automatic braking," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 10, no. 4, pp. 594 –605, dec. 2009.

[3] J. Hoc and S. Debernard, "Respective demands of task and function allocation on human-machine co-operation design: a psychological approach," *Connection Science*, 2002. [Online]. Available: http://www.informaworld.com/index/3NRP43BB47QV84GF.pdf

[4] M. Althoff, O. Stursberg, and M. Buss, "Online verification of cognitive car decisions," *2007 IEEE Intelligent Vehicles Symposium*, 2007.

[5] A. Heide and K. Henning, "The "cognitive car": A roadmap for research issues in the automotive sector," *Annual Reviews in Control*, vol. 30, no. 2, pp. 197–203, 2006. [Online]. Available: http://dx.doi.org/10.1016/j.arcontrol.2006.09.005

[6] E. Murphy-Chutorian and M. M. Trivedi, "Head pose estimation and augmented reality tracking: An integrated system and evaluation for monitoring driver awareness," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 11, no. 2, pp. 300 –311, june 2010.

[7] P. Balaji and D. Srinivasan, "Multi-agent system in urban traffic signal control," *Computational Intelligence Magazine, IEEE*, vol. 5, no. 4, pp. 43 –51, nov. 2010.

[8] P. Beeson, J. O'Quin, B. Gillan, T. Nimmagadda, M. Ristroph, D. Li, and P. Stone, "Multiagent interactions in urban driving," *Journal of Physical Agents: Multi-robot Systems*, vol. 2, no. 1, pp. 15–29, March 2008.

[9] S. Rodriguez Florez, V. Fremont, P. Bonnifait, and V. Cherfaoui, "Time to collision estimation using a multi-modal sensor fusion approach," in *IEEE Intelligent Vehicles Symposium*, San Diego (USA), June 2010, pp. 1–6.

[10] L. Edward, D. Lourdeaux, D. Lenne, J. Barthès, and J. Burkhardt, "Modelling autonomous virtual agent behaviours in a virtual environment for risk," *IJVR : International Journal of Virtual Reality*, vol. 7, no. 3, pp. 13–22, September 2008.

[11] C. Ress, A. Etemad, T. Hochkirchen, and D. Kuck, "Electronic horizon supporting adas applications with predictive map data," in *Intelligent Transportation Systems conference*, Hannover, 2005. [Online]. Available: http://www.ford.com/en/

[12] S. Durekovic and N. Smith, "Architectures of map-supported adas, booktitle = Intelligent Vehicles Symposium (IV), year = 2011, address = Baden-Baden, Germany, month = June 5-9,."

[13] O. Bezet, V. Cherfaoui, and P. Bonnifait, "A system for driver behavioral indicators processing and archiving," in *9th IEEE Conference on Intelligent Transportation Systems (ITSC 2006)*, 2006.

[14] J.-P. Barthès, "Omas-a flexible multi-agent environment for cscwd," *Future Generation Computing System*, no. 27, pp. 78–87, 2011.

[15] K. Chaaban, P. Crubillé, and M. Shawky, "A distributed framework for real-time in-vehicle applications," in *IEEE Conference on Intelligent Transportation Systems (ITSC)*, Vienne, Autriche, 13-16 september 2005.

[16] M. Shozakai, S. Nakamura, and K. Shikano, "Robust speech recognition in car environments," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 1, may 1998, pp. 269 –272 vol.1.

[17] F. Fayad and V. Cherfaoui, "Object-level fusion and confidence management in a multi-sensor pedestrian tracking system," in *Lecture notes in Electrical Engineering*, 2009, vol. 35.